Modelling Energy Consumption in Automated Vacuum Waste Collection Systems

Cèsar Fernández[a], Felip Manyà[b], Carles Mateu[a,*], Francina Sole-Mauri[c]

[a]*INSPIRES Research Institute, Universitat de Lleida, Lleida, Spain*
[b]*Artificial Intelligence Research Institute (IIIA, CSIC), Bellaterra, Spain*
[c]*Chemical and Biological Engineering Department, University of British Columbia, Vancouver, Canada*

Abstract

In a world where resources are scarce and urban areas consume the vast majority of these resources, it is vital to make cities greener and more sustainable. A smart city is a city in which information and communications technology are merged with traditional infrastructures, coordinated and integrated using new digital technologies. The increasing amount of waste generated, and the collection and treatment of waste poses a major challenge to modern urban planning in general, and to smart cities in particular. To cope with this problem, automated vacuum waste collection (AVWC) uses air suction on a closed network of underground pipes to transport waste from the drop off points scattered throughout the city to a central collection point, reducing greenhouse gas emissions and the inconveniences of conventional methods (odours, noise, etc.). Since a significant part of the cost of operating AVWC systems is energy consumption, we have developed a model with the aim of applying constraint programming technology to schedule the daily emptying sequences of the drop off points in such a way that energy consumption is minimized. In this paper we describe how the problem of deciding the drop off points that should be emptied at a given time can be modeled as a constraint integer programming (CIP) problem. Moreover, we report on experiments using real data from AVWC systems installed in different cities that provide empirical evidence that CIP offers a suitable technology for reducing energy consumption in AVWC.

*Corresponding author

Email addresses: `cesar@diei.udl.cat` (Cèsar Fernández), `felip@iiia.csic.es` (Felip Manyà), `carlesm@diei.udl.cat` (Carles Mateu), `francina.sole@gmail.com` (Francina Sole-Mauri)

Keywords:
Waste Collection, Smart City, Artificial Intelligence, Constraint Integer Programming (CIP), Sustainability, Computational Sustainability

---

1. Introduction

Awareness has arisen that cities have to develop in a greener and more sustainable way, since they consume the majority of the world resources. Urban areas face both a changing urban population size and increasing sustainability issues in terms of providing good socioeconomic and environmental living conditions. Urban planning has to deal with both challenges ([16]).

A new city model has been proposed in the last years ([7, 22, 18, 26, 38]), called smart city, which represents a community of average technology size, interconnected and sustainable, comfortable, attractive and secure. A smart city is a city in which information and communication technologies are merged with traditional infrastructures, coordinated and integrated using new digital technologies. Advanced systems to improve and automate processes within a city will play a leading role in smart cities. From smart design of buildings to intelligent control systems the possible improvements enabled by sensing technologies are immense. Two of the critical problems related to cities are transport and energy ([25]). The cities consume 75% of worldwide energy production and generate 80% of CO_2 emissions.

Considering the growth of urban population, along with the increasing amount of waste generated on wealth areas, the collection and treatment of waste poses a major challenge on modern urban planning ([15]). Waste generation per inhabitant has been increased for years and, according to last projections, this will continue till 2020. The environmental issues related to waste collection are believed to be related primarily to the use and combustion of diesel in vehicles, because the emission of exhaust gases from the combustion process. In a broader context environmental issues are also related to noise and odor ([21]).

Automated vacuum waste collection (AVWC) uses air suction on a closed network of underground pipes, covering an area of a few square kilometers, to transport waste from the drop off points scattered throughout the city to a central collection point, reducing greenhouse gas emissions and the inconveniences of the conventional method of waste collection (odors, noise, combustion gas emissions, etc.), as well as allowing better waste reuse and recycling. An AVWC system has a control software that includes the implementation of a method for deciding when and which inlets, corresponding to the same fraction, should be emptied during a time interval taking into account a number of constraints (e.g., full inlets should always be emptied, inlets should be emptied at least once a day, air speed, …). The use of artificial intelligence and other IT techinques can lead to more efficient energy consumption by defining smarter daily operation plans, and increasing the environmental sustainability ([19]).

Optimization of conventional methods for waste collection, using trucks to pick up waste from collecting points scattered throughout the city has received considerable attention ([17]). [32] proposed a benchmark problem set with time windows for the vehicle routing problem, building a route one at a time in a serial manner. A computational study of several heuristic algorithms was conducted, and an insertion heuristic, which is known as Solomon's insertion heuristic, generates the best routing schedules in many cases. [30] built routes in parallel using Solomon's heuristic to find the initial seed customers and number of vehicles. [31] generated multiple initial solution by a local search heuristic. [33] used a new edge exchange heuristic and tabu search to improve vehicle routing solutions. Multiple use of vehicles in which multiple routes can be served by the same vehicle were considered. [14] used a Markov decision process to model the residential waste collection problem in the city of Chicago. [35] modified Solomon's insertion algorithm and applied it to a waste collection problem in Hanoi, Vietnam. [29] used several non-traditional constraints of real-life vehicle routing problems, such as multiple capacity, vehicle type, region served first or last constraints. A savings-based method is proposed for the problems that have those constraints. [5, 6] addressed the periodic vehicle routing problem with intermediate facilities, using a tabu search algorithm. [34] also used an heuristic approach for the separate collection of three types of waste. [8] applied ant colony optimization to solve the urban waste collection problem in a city of the metropolitan area of Barcelona; [11] solved the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities using tabu search and variable neighborhood search; [20] addressed the same problem as [11] but their solution relies on extending Solomon's insertion algorithm to this new context; and [4] defined a network model for waste collection in Northern Cyprus which produced considerable cost savings.

The problem of optimizing AVWC systems to reduce energy usage has not been studied so far, despite a significant part of the cost of operating AVWC systems is energy consumption. This paper establishes a first step in this direction by formally defining AVWC systems, modeling the problem of scheduling the drop off points that should be emptied at a given time, taking into account energy consumption, as a constraint integer programming (CIP) problem [1, 2], and solving it in real-time. Moreover, we report on experiments using real data from AVWC systems installed in different cities that provide empirical evidence that CIP offers a suitable technology for reducing energy consumption in AVWC. These results pave the way for developing an scheduler capable of producing optimal plans of the daily emptying sequences of the drop off points in such a way that energy consumption is minimized.

A Constraint programming approach has been adopted because constraint programming is a powerful technology for solving hard combinatorial problems in a diverse range of application domains, including complex real-world planning and scheduling problems; see for example [24, 23] for the application of constraint programming for solving real-world operation problems over oil pipeline networks. Its solving techniques have their roots in artificial intelligence, mathematical programming, operations research, and programming languages, and benefits from the experience of all these research communities [27].

The paper is structured as follows. First, AVWC systems are formally described, as well as the problem dynamics, giving expressions for the time and energy calculations that define the system operation. Then the problem is defined, scheduling the daily emptying sequences of the drop off points in such a way that energy consumption is minimized. An example of the system dynamics is provided and the CIP model is presented. Finally, an empirical investigation on real-world instances is reported, analyzing the obtained results. To conclude, we present our current ongoing work on using our CIP approach for solving the more general dynamic problem, when one has to optimize the selected operations over a full period of time. This paper is an extended version that completes the work presented in [10, 9]. In [10], we formally define the problem of optimizing energy in AVWC systems, and in [9], we describe the solving approach we used to solve that problem. Here, we provide a more detailed explanation of the problem and of the solving approach by clarifying the main concepts with additional examples,incorporating the topology of a real-world AVWC plant that operates in a Spanish city, showing step by step the operations performed by an AVWC plant for emptying a sequence of inlets, improving the explanation of how inlets are ordered, making publicly available a wider range of data sets, and adding a list of acronyms and symbols.

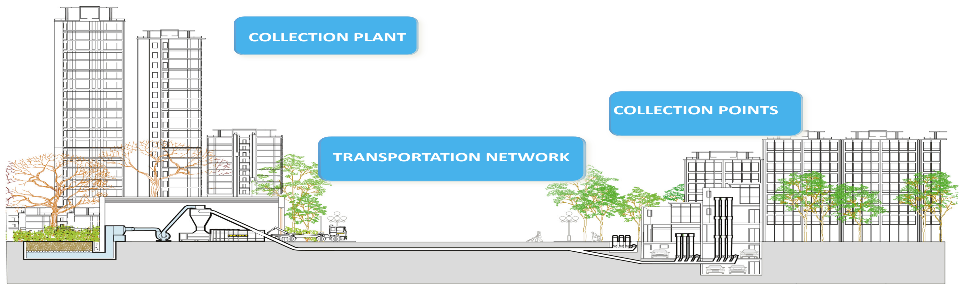

Figure 1: Scheme of an automated vacuum waste collection system

2. System description

In AVWC the pipe network has a tree shape, and the central collection point is located at the root node. This central collection point has the means to split the collected waste by fraction (organic fraction, paper, etc.), and is where waste is packed for disposal in containers that are then transported with trucks to a landfill area for recycling or Mechanical Biological Treatment. The network usually has sector valves located on some of the branch junctions that can isolate one of the branches (to reduce the volume of air that will be suctioned). The drop off points are located along the branches, and contain inlets for the different fractions. There are also air valves that act as air entry points that help produce the air flow when the suction starts. Air valves can be located next to inlets, although it is not mandatory to have an air valve in each drop off point. Figure 1 shows a general view of an AVWC system.

An underground vacuum waste collection system is modeled as a set $\{\mathcal{T}, \mathcal{I}, \mathcal{F}, \mathcal{V}^a, \mathcal{V}^s\}$. $\mathcal{T}(\mathcal{N}, \mathcal{E})$ is a rooted binary tree with nodes ($\mathcal{N}$) representing either waste inlets ($\mathcal{I}$) or pipe junctions, and edges ($\mathcal{E}$) corresponding to union pipes between nodes. $\mathcal{F}$ represents the set of fractions waste is divided into. Typical fractions include organic refuse, paper, plastic and glass. Air valves ($\mathcal{V}^a$), located at some inlets, create air streams able to empty downstream inlets. Sector valves ($\mathcal{V}^s$) are disposed along the tree in order to segment the whole tree struc-

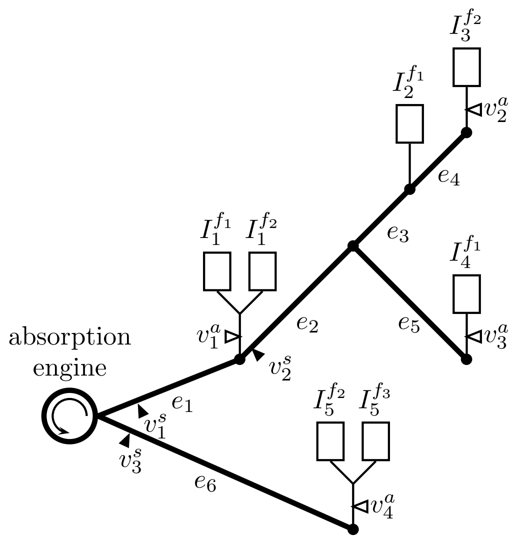

Figure 2: Schematic example of an automatic vacuum waste collection plant

ture, defining isolated sectors (s), making a more efficient transport for the inlets comprised in the corresponding sector. The sector, defined by a configuration of open and closed sector valves, is the subtree that contains all the paths to the root that contain only open valves and at least an inlet.

Each inlet in $\mathcal{I}$ is denoted by I_i^f, where i denotes the inlet number and f denotes the fraction, meanwhile v_i^a and v_i^s denote air and sector valves respectively. The status of any valve is open (o) or closed (c). Fig. 2 is a small example of the system, with 3 types of fraction, 5 inlets (two of them handling 2 types of fraction, so one can consider having 7 inlets), 4 air valves and 3 sector valves. Note that, in this case, only 5 combinations of $\mathcal{V}^s$ out of the 8 possible are valid, giving 5 different sectors, as depicted in Fig. 3.[1]

Three important subtrees that will deeply impact the system dynamics arise from the topology: emptying, air and vacuum subtrees. The emptying subtree ($\mathcal{T}_i^E$) is unique for each inlet, and is defined as the path that waste must follow from inlet i to the root node. Of course, $\mathcal{T}_i^E$ must not contain closed sector valves

[1]Following the notation (v_1^s, v_2^s, v_3^s), $\{(\mathrm{c},\mathrm{c},\mathrm{c}),(\mathrm{c},\mathrm{o},\mathrm{c})\}$ are not valid assignments (because the resulting subtree only contains the root node) and $\{(\mathrm{c},\mathrm{c},\mathrm{o}),(\mathrm{c},\mathrm{o},\mathrm{o})\}$ give the same sector configuration.

on it. The air subtree ($\mathcal{T}_i^A$) is the path followed by the air stream in charge of waste transport along $\mathcal{T}_i^E$. Note that $\mathcal{T}_i^E \subseteq \mathcal{T}_i^A$, being equal if inlet i has an air valve, otherwise, the airflow must come from an upstream inlet. The vacuum subtree ($\mathcal{T}_s^V$) is unique for each sector and represents the total amount of air to be moved before proceeding to waste transport. Let's denote by $d(\mathcal{T})$ the total length of a tree.

As an example, let's consider inlet number 2. In this case, $d(\mathcal{T}_2^E) = d(e_1) + d(e_2) + d(e_3)$ and $d(\mathcal{T}_2^A) = d(\mathcal{T}_2^E) + d(e_4)$. Inlet 2 can be emptied by 2 sectors; $s_{1,2,\overline{3}}$ and $s_{1,2,3}$ ($v_1^s =$ o, $v_2^s =$ o, $v_3^s =$ c and $v_2^s =$ o, $v_3^s =$ o respectively). For the first case, $d(\mathcal{T}_{s_{1,2,\overline{3}}}^V) = d(e_1) + d(e_2) + d(e_3) + d(e_4) + d(e_5)$, and for the second case $d(\mathcal{T}_{s_{1,2,3}}^V) = d(e_1) + d(e_2) + d(e_3) + d(e_4) + d(e_5) + d(e_6)$.

Consider now the waste occupancy L_i^f of an inlet I_i^f at the beginning of the current slot of time t. As we will discuss next, the energy consumption depends on the sequence of inlets we decide to (completely) empty [2]. Given the selected sector s and fraction f for the current time t, a valid emptying sequence $\mathcal{E}_t^{f,s} = [I_{i_1}^f, I_{i_2}^f, \ldots]$ is an ordered subset of the inlets in sector s such that the total waste to be emptied does not exceed a maximum transfer capacity (L_{max}^f) for fraction f, that is: $\sum_{I_i^f \in \mathcal{E}_t^{f,s}} L_i^f \leq L_{max}^f$. Emptying sequences can not overlap in time and can be null (do not empty any inlets at this time).

3. Dynamics of the model

Energy consumption depends on the emptying sequence selected for the current time slot and on other operational attributes of the system such as air speed, type of fraction and section selected to be emptied, inlets ordering, ... Air speed operation [37] is an important one, being crucial to determine the duration of an emptying sequence and, consequently, energy consumption. For our model, we will assume that we operate at a constant air speed during an emptying sequence (v_t). Due to structural reasons, v_t has a maximum (V_M). Furthermore, each inlet is characterized by a minimum air speed operation (V_i^f) to avoid pipe obstructs.

The second element is the operation time (T_t). It is defined as the required time to operate an emptying sequence, depending on the sequence itself, the air speed of operation v_t, and the previous operation state of the system. Such a previous operation state can be: operating an emptying sequence for type of fraction f' and sector s' at speed v_{t-1} or idle ($v_{t-1} = 0$).

[2]Partial emptying of inlets is not considered in our current model

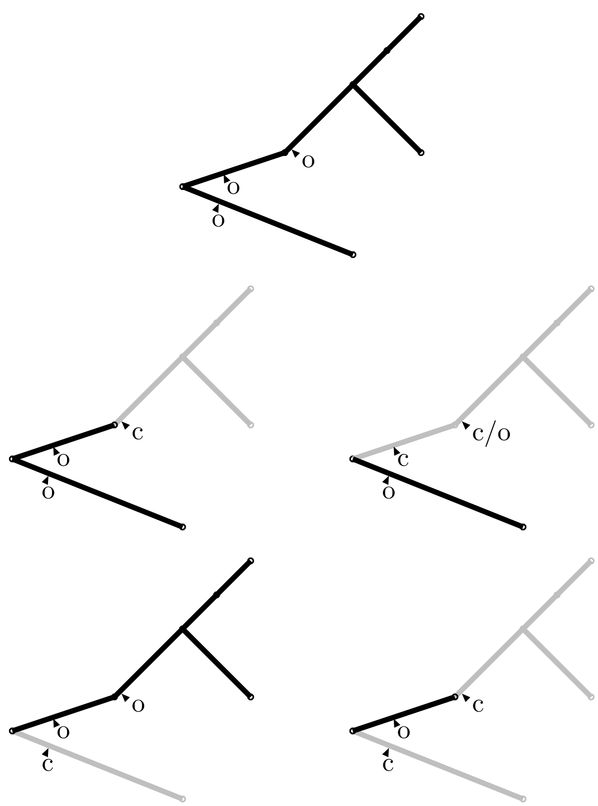

Figure 3: Schematic representation of all the possible sectors according to the sector valves set up.

The operation time is divided into two phases: transitory and stationary. In the transitory phase the previous speed (v_{t-1}) changes progressively to v_t in time T_t^{tr} and next in the stationary phase the selected emptying sequence is executed in time T_t^{st}. T_t^{tr} is a function of three types of parameters. First, the previous and the current operational air speed. Second, the type of fraction, because if there is a change of type of fraction among the previous and actual emptying sequences, the air speed must be dropped to a low value due to operational requirements. Otherwise, it is enough to increase or decrease the air speed from the previous value (v_{t-1}) to the actual (v_t). Third, the total amount of air to be adapted. It depends on the vacuum paths of the previous sector (s') and the current sector (s), and can be obtained as $\mathcal{T}_s^V - \mathcal{T}_s^V \cap \mathcal{T}_{s'}^V$. We can express

$$T_t^{tr} = \begin{cases} c_{1,t}^{tr} \cdot |v_t - v_{t-1}| \\ +c_{2,t}^{tr} \cdot \left(d(\mathcal{T}_s^V) - d(\mathcal{T}_s^V \cap \mathcal{T}_{s'}^V) \right), & f = f', \\ c_{1,t}^{tr} \cdot (v_t + v_{t-1}) \\ +c_{2,t}^{tr} \cdot \left(d(\mathcal{T}_s^V) - d(\mathcal{T}_s^V \cap \mathcal{T}_{s'}^V) \right), & f \neq f', \end{cases}$$

where $c_{1,t}^{tr}$ and $c_{2,t}^{tr}$ are constants for a given system, as detailed in the following section.

Once the transitory phase ends and the new air speed is reached, the stationary phase can be started in order to proceed with the emptying sequence. An emptying sequence consists of two operations that iterate over the ordered sequence of inlets; first, to empty an inlet over the transport pipes, and second, to proceed to waste transport. The transport of waste and the emptying phase of the next inlet can overlap in time, if and only if, the inlet to be emptied is upstream the estimated position of the waste being transported. Under these assumptions, given the current emptying sequence $\mathcal{E}_t^{f,s}$ and current inlet loads (L_i^f) we have that:

$$T_t^{st} = \sum_{\substack{I_i^f \in \mathcal{E}_t^{f,s} \\ I_j^f = \text{next}(I_i^f)}} T_t^{st}(i,j),$$

$$T_t^{st}(i,j) = c_{1,t}^{st} \cdot L_{i,t}^f + \frac{d(\mathcal{T}_i^E) - d(\mathcal{T}_i^E \cap \mathcal{T}_j^E)}{v_t}, \tag{1}$$

where $next(I_i^f)$ denotes the following element to I_i^f in the ordered sequence $\mathcal{E}_t^{f,s}$, and $next()$ of the last element in the sequence is the root node. $T_t^{st}(i,j)$ is obtained as the sum of the time needed for emptying the inlet I_i^f (left term of the sum) with the time needed to transport its load down to the intersection point with the emptying path $\mathcal{T}_j^E$ of the next inlet I_j^f. Note that if I_j^f is upstream I_i^f, then

$$d(\mathcal{T}_i^E) - d(\mathcal{T}_i^E \cap \mathcal{T}_j^E) = 0,$$

meaning that once emptied I_i^f, we can proceed to empty I_j^f.

The last element that defines the dynamics of our model is energy. Energy is closely related to the operation time, and can also be split into two parts; transitory and stationary. It is easy to understand that, for the transitory case, there is only energy consumption for the process of increasing air speed but not for decreasing it. That being so, we can write

$$E_t^{tr} = \begin{cases} c_{1,e}^{tr} \cdot |v_t - v_{t-1}|^+ \\ +c_{2,e}^{tr} \cdot \left(d(\mathcal{T}_s^V) - d(\mathcal{T}_s^V \cap \mathcal{T}_{s'}^V)\right), & f = f', \\ c_{1,e}^{tr} \cdot v_t \\ +c_{2,e}^{tr} \cdot \left(d(\mathcal{T}_s^V) - d(\mathcal{T}_s^V \cap \mathcal{T}_{s'}^V)\right), & f \neq f', \end{cases}$$

where

$$|x|^+ = \begin{cases} 0 & x < 0, \\ x & x \geq 0. \end{cases}$$

For the stationary part of the energy, the air path plays an important role. For the same emptying path, the minimum transport energy is obtained when the shortest air path is employed, that is, opening the upstream air valve closest to the inlet being emptied. The type of fraction also affects the power requirements because denser fractions require more energy. Under these considerations, we assume that during the stationary phase, power consumption is a linear function of the air path, $c_{1,e}^{st}(f) + c_{2,e}^{st}(f) \cdot d(\mathcal{T}_i^A)$, with coefficients depending on the type of fraction. Stationary energy results,

$$E_t^{st} = \sum_{\substack{I_i^f \in \mathcal{E}_t^{f,s} \\ I_j^f = \text{next}(I_i^f)}} \left(c_{1,e}^{st}(f) + c_{2,e}^{st}(f) \cdot d(\mathcal{T}_i^A)\right) \cdot T_t^{st}(i,j). \tag{2}$$

That is, the contribution to the stationary energy of an inlet I_i^f of the emptying sequence is proportional to its air path ($\mathcal{T}_i^A$) and to its transport time up to the next intersection.

4. Problem description

The objective of the problem is to find a set of emptying sequences and air speed operations, $\{\mathcal{E}_t^{f,s}\} \times \{v_t\}$, $0 \leq t \leq T$, for an operative period of time T

(e.g. a day), that minimizes the energy cost, $\sum_{t=0}^{T} f_c(t) \cdot (E_t^{tr} + E_t^{st})$, where $f_c(t)$ is the energy cost function. Also, at the end of the period of time T, the residual load L_i^f of inlet I_i^f should be below a lower bound ϵ_i^f.

In this article we are dealing only with the problem of selecting an optimal emptying sequence for the state of the system on a given time slot t, leaving as future work the problem of optimizing over a full period of time T, i.e. the dynamic continuous problem. Once this is taken into account, the problem at time t will be subject to the following hard constraints and conditions:

- The emptying sequence $\mathcal{E}_t^{f,s} = [I_{i_1}^f, I_{i_2}^f, \ldots]$ must be valid. That is: $\sum_{I_i^f \in \mathcal{E}_t^{f,s}} L_i^f \leq \underline{L_{max}^f}$
- The air speed satisfies its operational range: $\max_{I_i^f \in \mathcal{I}^s}(\underline{V_i^f}) \leq v_t \leq \underline{V_M}$, $0 < t \leq T$.
- Any inlet I_i^f with a load L_i^f over a threshold $\underline{\mathrm{th}_i^f}$ should be included in the emptying sequence. Given that it may not be possible to include all the inlets that are overloaded, we will try to force the inclusion of the maximum number of such inlets by a penalty term in our objective function.

The problem is described by three types of files. A first set of files (`topology.xml`) is used to encode the network topology of the problem (i.e. edges, valves, inlet location, etc.) as well as to provide initial inlet load and inlet specific parameters and constants.

A second set of files (`parameters.xml`) details the constants of the dynamics model ($c_{1,t}^{tr}, \cdots$), as well as the energy cost ($f_c(t)$) depending on time and according to the energy fares. The energy cost function is expressed as a look-up table in terms of price per energy unit depending on date/time. This set of files also contains the constant values defined in the three types of constraints detailed above (constants appear underlined), such as maximum laod capacity per fraction, air speed ranges and residuald thresholds.

Finally, a third set of files (`data.xml`) describes the stochastic component of the system, that is, the way that the users dispose waste into the inlets, during a period of time (usually several weeks). This last set of files describes the process of disposal volumes, it can give either a list of real world disposals or a parametrized random function for arrival times and waste amount for any inlet.

5. Example

In this section we give a small example of the dynamics of the system based on the topology of Fig. 2. Let's assume the following, as shown in figure 4a:

- Before $t = 0$, the system is not operating ($v_{0^-} = 0$).
- At $t = 0$, we decide to empty fraction f_1 from inlets $I_1^{f_1}$, $I_2^{f_1}$ and $I_4^{f_1}$, through section $s_{1,2,\overline{3}}$, with air speed v_0.
- Inlets load at $t = 0$ is $L_1^{f_1} + L_2^{f_1} + L_4^{f_1} \leq L_{max}^{f_1}$.
- The emptying sequence is ordered as; $[I_2^{f_1}, I_4^{f_1}, I_1^{f_1}]$. Section 6 details the ordering process.

The chronological events resulting from such a decision are the following:

1. At $t = 0$ the system starts the transitory phase to speed up the air volume of the operating section from 0 to v_0. This will take $T_0^{tr} = c_{1,t}^{tr} \cdot v_0 + c_{2,t}^{tr} \cdot (e_1 + e_2 + e_3 + e_4 + e_5)$ seconds. See Fig. 4b.
2. At $t = T_0^{tr}$ we start the stationary phase. This phase will consist of:
   - To empty $I_2^{f_1}$, taking $c_{1,t}^{st} \cdot L_2^{f_1}$ seconds. See Fig. 4c.
   - To transport waste down to the next junction (along edge e_3). This will take e_3/v_0 seconds. Once the load $L_2^{f_1}$ is downstream the junction, the inlet $I_4^{f_1}$ can be started to be emptied. See Fig. 4d.
   - To empty $I_4^{f_1}$, taking $c_{1,t}^{st} \cdot L_4^{f_1}$ seconds. See Fig. 4e.
   - To transport waste down to next inlet (along edge e_5 and e_2). This will take $(e_5 + e_2)/v_0$ seconds. See Fig. 4f and Fig. 4g.
   - To empty $I_1^{f_1}$, taking $c_{1,t}^{st} \cdot L_1^{f_1}$ seconds. See Fig. 4h.
   - To transport waste down to the root node (along edge e_1). This will take e_1/v_0 seconds. See Fig. 4i.
3. At this point, the transport is finished and we can decide for the next emptying sequence, and the system is in the state shown in Fig. 4j.

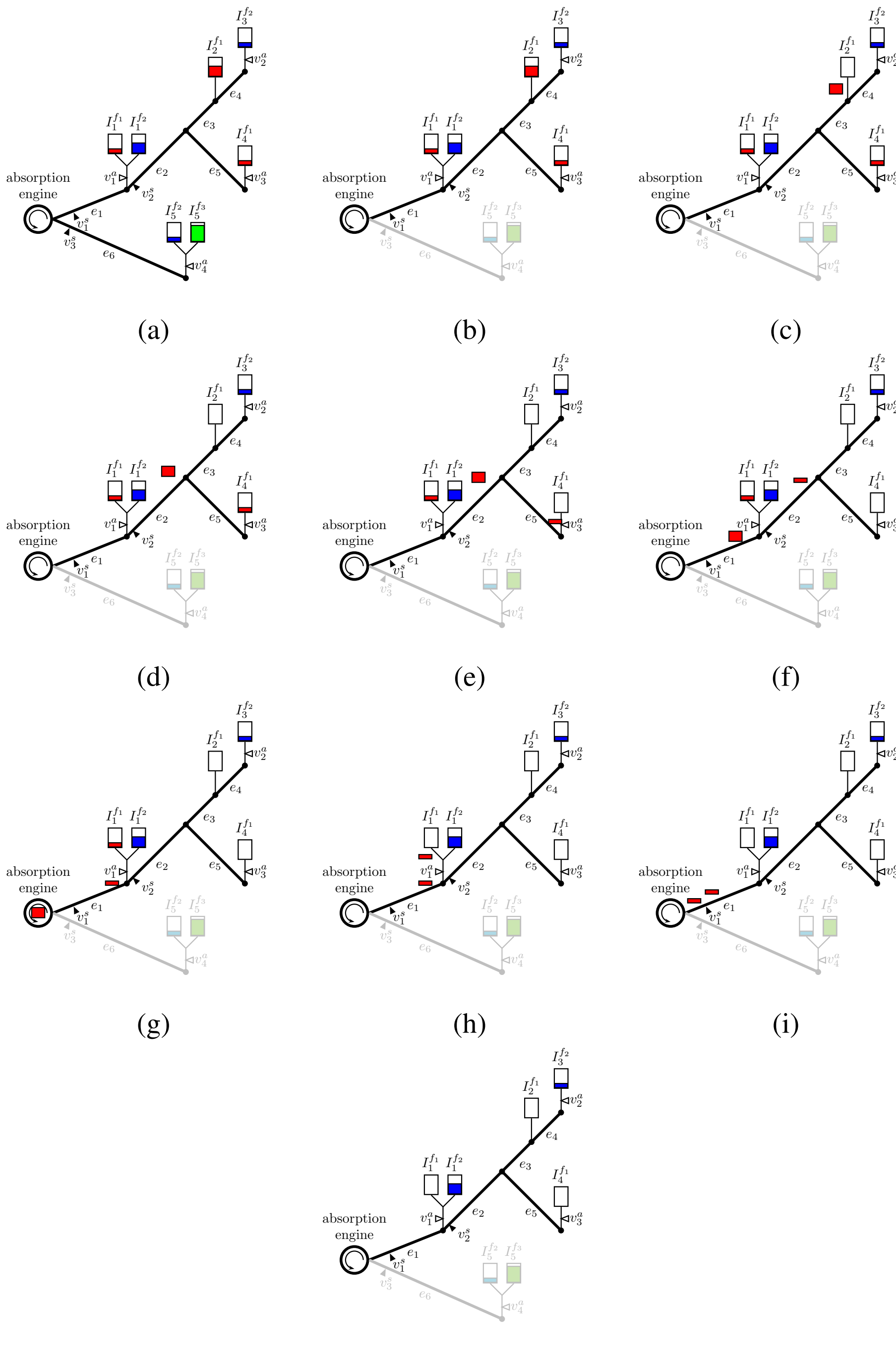

Figure 4: Operating example of an AWCS plant

6. CIP Encoding

Constraint Integer Programming (CIP) is a novel paradigm that integrates constraint programming (CP), mixed integer programming (MIP), and satisfiability (SAT) modeling and solving techniques [1, 2, 3]. CIP can be seen as a generalization of MIP that allows for inclusion of arbitrary constraints that get reduced to linear constraints on continuous variables after all integer variables have been fixed. CIPs can be treated by a combination of techniques used to solve CPs, MIPs, and SAT problems: propagating the variable domains by constraint specific algorithms, solving a linear programming (LP) relaxation of the problem, strengthening the LP by cutting plane separation, and analyzing infeasible subproblems to infer useful global knowledge about the problem instance. The motivation behind combining CP, SAT, and MIP techniques is to combine their advantages and to compensate for their individual weaknesses.

In this section we describe a CIP encoding for the problem of determining an optimal emptying sequence $\mathcal{E}_t^{f,s}$ and an optimal operational air speed v for the current time slot. Before solving the encoding of the problem, there is a first phase where we obtain a total ordering for the whole set of inlets $\mathcal{I}$. Then, in the second phase, we perform a search for an optimal subset of inlets, subject to the ordering found in the first phase.

As reflected in Eq. 2, the order in which inlets are emptied and how their corresponding air valves are operated, determines the stationary energy. The objective of determine a previous ordering is twofold. First, to find the ordering emptying sequence among the complete set of inlets that minimizes E_t^{st}, as well as the number of air valve switches, independently of the inlets load. It must be considered that such an optimal ordering applies for every system state and slot time. Consequently, the second objective avoids the CIP solver to decide the ordering, reducing this way the search space.

We first describe the ordering algorithm in the next subsection and then the Constraint Integer Programming (CIP) model used for solving the problem. A CIP model is defined by a set of variables (integer and real valued), constraints (among the variables) and an objective function.

Ordered inlets

We present here the algorithm for ordering once all the inlets of the plant, and then the resulting total order is used for ordering any subset of inlets chosen at any execution of the optimization algorithm. Although ordering all the inlets

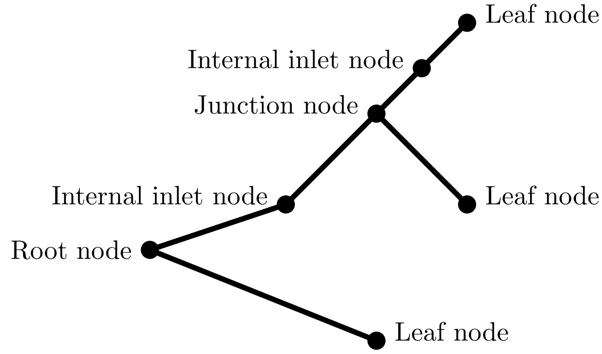

Figure 5: Types of nodes example for the inlets ordering algorithm

before the search phase is not necessarily the best possible option, in practice it is a suitable option to obtain good quality results within reasonable time.

The notation and auxiliary functions of the algorithm are:

- $\mathcal{I}$: It is the set of inlets.
- $\mathcal{T}(\mathcal{N}, \mathcal{E})$: It is the binary tree where nodes can be of different classes (Fig. 5 exemplifies the node classes detailed below):
  - Root node: It is the node with the absorption engine, and is connected to either a junction node or an internal inlet node.
  - Junction node: It is an internal node with two upstream pipes and one downstream pipe. We denote the two son nodes of a junction node n as son1(n) and son2(n).
  - Internal inlet node: It is an internal node with an inlet which may contain also an air valve, a downstream pipe and an upstream pipe. We denote the son (upstream) on an internal inlet node n as son(n).
  - Leaf node: It contains an inlet with its corresponding air valve.
- getfurthest($\mathcal{T}(\mathcal{N}, \mathcal{E})$, r): get the furthest inlet node fn of the subtree $\mathcal{T}_r$, and its distance d, from the root of $\mathcal{T}(\mathcal{N}, \mathcal{E})$ in the pair (fn, d).

The ordering function (see Algo. 1) is defined recursively for a plant tree, or plant subtree, and the ordering it gives depends on the following possible cases for the plant tree:

1. Leaf inlet node: in this base case the ordering contains only the inlet node.

Algorithm 1: Function **OrderInlets** ($\mathcal{T}(\mathcal{N},\mathcal{E})$) recursively defined

```
L := [] ;
// L will contain the ordered list of inlets, starting
   from the furthest node from rnode up to rnode
;
switch typeof(rnode) is do
    case junction
        (f1, d1) := getfurthest( T(N, E), son1(rnode) ) ;
        (f2, d2) := getfurthest( T(N, E), son2(rnode) ) ;
        L1 := OrderInlets ( T(N, E), son1( rnode ) ) ;
        L2 := OrderInlets ( T(N, E), son2( rnode ) ) ;
        if d1 ≥ d2 then
            L := L1|L2
        else
            L := L2|L1
    case inner inlet
        L := OrderInlets ( T(N, E), son( rnode ) ) |[rnode]
    case leaf node
        L := [rnode]
return L ;
```

2. Internal inlet node: all the inlet nodes of the subtree attached to the node will be inserted first in the ordering, and then the internal inlet node.
3. Junction node: all the inlet nodes of the subtree that contains the furthest node from the root node will be inserted first, and then the inlet nodes of the other subtree.

The reasons behind such an ordering are that for the energy needed in the stationary phase we have the following behavior. Consider the simplest case, a single branch with n inlets and the root node, being I_1^f the nearest inlet to the root (directly connected to it with an edge) and being I_n^f the furthest inlet. In this case, we assume the following:

- All the coefficients in Eq. 1 and Eq. 2, as well as v_t are equal to one.
- All the inlets have an air valve, so the inlet air path coincides with the inlet emptying path, $\mathcal{T}_i^A = \mathcal{T}_i^E$.
- Note that the inlet unload time in Eq. 1 is not considered, because its contribution to E_t^{st} does not depend on the inlet ordering.

We consider the stationary energy for two orderings, a first ordering by ascending distance to the root ($E_{t,a}^{st}$) and a second one by descending distance ($E_{t,d}^{st}$). For the ordering by ascending distance to the root ($[I_1^f, I_2^f, \ldots, I_n^f]$), denoting as d_i the distance between I_i^f and I_{i-1}^f, we have

$$\begin{aligned} E_{t,a}^{st} &= O\left(\left(1+\sum_i d_i\right)\sum_i d_i\right) \\ &= O\left(\sum_i d_i + \left(\sum_i d_i\right)^2\right). \end{aligned}$$

That is, the energy is dominated by the square of the length of the longest air path, because for emptying all the inlets we are operating over the longest air path. Conversely in the ordering by descending distance ($[I_n^f, I_{n-1}^f, \ldots, I_1^f]$) we found that the spent stationary energy is less because, in this case, we have:

$$\begin{aligned} E_{t,d}^{st} &= O\left(\sum_i \left(1+\sum_{j\le i} d_j\right) d_i\right) \\ &= O\left(\sum_i d_i + \sum_{j\le i} d_i \cdot d_j\right) < E_{t,a}^{st}. \end{aligned}$$

Parameters and functions

For presenting our CIP model, we first introduce some constants and parameters. From the system model $\{\mathcal{T}, \mathcal{I}, \mathcal{F}, \mathcal{V}^a, \mathcal{V}^s\}$, the possible dispositions of the sector valves ($\mathcal{V}^s$) determine a set of sectors ($\mathcal{S}$).

- $pu_i^f \in \mathbb{R}, f = 1 \dots |\mathcal{F}|, i = 1 \dots |\mathcal{I}|$, is the penalty cost due to leaving unloaded inlet I_i^f above a given threshold of its maximum capacity (th_i^f).
- $L_i^f \in \mathbb{R}$, is the load of I_i^f at the beginning of the slot time being optimized ($I_{i,t}^f$).
- $s' \in [1 \dots |\mathcal{S}|]$, denotes the previous active sector.
- $f' \in [1 \dots |\mathcal{F}|]$, denotes the previous active fraction.
- $v' \in \mathbb{R}$, denotes the previous operational speed.
- $\mathrm{Ind} : \{0 \dots |\mathcal{I}|\} \rightarrow \{0 \dots |\mathcal{I}|\}$, is a function that orders the set $\mathcal{I} \cup \emptyset$ according to the above definition, being $\emptyset$ the root node.
- $\mathrm{Vp} : \{(1 \dots |\mathcal{S}|) \times (1 \dots |\mathcal{S}|)\} \rightarrow \mathbb{R}\}, \mathrm{Vp}(s, s') = d(\mathcal{T}_s^V) - d(\mathcal{T}_s^V \cap \mathcal{T}_{s'}^V)$, is the vacuum path.
- $\mathrm{Ep} : \{(0 \dots |\mathcal{I}|) \times (0 \dots |\mathcal{I}|)\} \rightarrow \mathbb{R}\}, \mathrm{Ep}(i, j) = d(\mathcal{T}_i^E) - d(\mathcal{T}_i^R \cap \mathcal{T}_j^E)$, is the emptying path.
- $\mathrm{Ap} : \{1 \dots |\mathcal{I}|\} \rightarrow \mathbb{R}\}, \mathrm{Ap}(i) = d(\mathcal{T}_i^A)$, is the air path.

Variables

In our CIP model, we have decision variables, over which the search is performed, and auxiliary variables in order to obtain a more readable set of constraints. We define the following decision variables:

- $s_i \in \{0, 1\}, i = 1 \dots |\mathcal{S}|$, indicates whether a sector is activated for emptying.
- $f_i \in \{0, 1\}, i = 1 \dots |\mathcal{F}|$, determines the fraction.
- $l_i^{s,f} \in \{0, 1\}, s = 1 \dots |\mathcal{S}|, f = 1 \dots |\mathcal{F}|, i = 1 \dots |\mathcal{I}|$, indicates whether inlet I_i^f is going to be emptied with sector s.
- $v \in \mathbb{R}$. Operational air speed.

The auxiliary variables, with its defining constraints, are:

- $b_i^f = \sum_{s=1}^{|\mathcal{S}|} l_i^{s,f} \in \{0,1\}, f = 1 \dots |\mathcal{F}|, i = 1 \dots |\mathcal{I}|$, indicates whether an inlet I_i^f is active regardless of the sector being considered.
- $p_{i,j}^{s,f} \in \{0,1\}, s = 1 \dots |\mathcal{S}|, f = 1 \dots |\mathcal{F}|, i = 0 \dots |\mathcal{I}|, j = 1 \dots |\mathcal{I}|, \mathrm{Ind}(i) < \mathrm{Ind}(j)$. It is a pair of inlet indicators, keeping track active inlets and their corresponding following up. It helps to compute energy.
- $v_{tr} \in \mathbb{R}$, is the air speed value to be considered in the transitory phase for energy calculation, depending on the fraction to be transported.
- $E_{tr} = c_{1,e}^{tr} \cdot v_{tr} + c_{2,e}^{tr} \cdot \sum_{i=1}^{|\mathcal{S}|} s_i \cdot \mathrm{Vp}(i, s') \in \mathbb{R}$, is the transitory energy.
- $T_{i,j}^{st} = \sum_{s=1}^{|\mathcal{S}|} \sum_{f=1}^{|\mathcal{F}|} \left(c_{1,t}^{st} \cdot l_i^{s,f} \cdot L_i^f + \frac{p_{i,j}^{s,f} \cdot \mathrm{Ep}(j,i)}{v} \right) \in \mathbb{R}, i = 1 \dots |\mathcal{I}|, j = 1 \dots |\mathcal{I}|$, is the stationary time.
- $c_{i,e}^{st} = \sum_{j=1}^{|\mathcal{F}|} c_{i,e}^{st}(j) \cdot f_j \in \mathbb{R}, i \in \{1,2\}$, determines the coefficients for the stationary energy calculation.
- $E_{st} = \sum_{i=1}^{|\mathcal{I}|} \sum_{\substack{j=1 \\ \mathrm{Ind}(i)<\mathrm{Ind}(j)}}^{|\mathcal{I}|} \left(c_{1,e}^{st} + c_{2,e}^{st} \cdot \mathrm{Ap}(j) \right) T_{i,j}^{st} \in \mathbb{R}$, is the stationary energy.
- $P = \sum_{i=1}^{|\mathcal{I}|} \sum_{f=1}^{|\mathcal{F}|} pu_i^f \cdot (1 - b_i^f) \in \mathbb{R}$, is the penalty due to leave unloaded inlets above a given threshold.

Constraints

The main constraints in our encoding are:

- $\sum_{i=1}^{|\mathcal{S}|} s_i \leq 1$. At most one active sector.
- $\sum_{i=1}^{|\mathcal{F}|} f_i \leq 1$. At most one active fraction.
- $\neg s_j \Rightarrow \neg l_i^{j,f}, \forall i = 1 \dots |\mathcal{I}|, j = 1 \dots |\mathcal{S}|, f = 1 \dots |\mathcal{F}|$. Inactive sector propagation.
- $\neg f_k \Rightarrow \neg l_i^{s,k}, \forall i = 1 \dots |\mathcal{I}|, s = 1 \dots |\mathcal{S}|, k = 1 \dots |\mathcal{F}|$. Inactive fraction propagation.
- $\sum_{i,s,f} l_i^{s,f} \cdot L_i^f \leq L_{max}^f$. Maximum transfer load per fraction.

- $\max_{i=1\ldots|\mathcal{I}|}(V_i^f \cdot b_i^f) \leq v \leq V_M \ \in \mathbb{R}$. Operational air speed.
- $\sum_{i=1}^{|\mathcal{I}|} p_{0,i}^{s,f} \leq 1, \forall s = 1 \ldots |\mathcal{S}|, f = 1 \ldots |\mathcal{F}|$. Root node ($\emptyset$) has at most one follow up inlet.
- $\neg l_i^{s,f} \Rightarrow \left(\neg p_{i,j}^{s,f} \wedge \neg p_{j',i}^{s,f}\right), \forall s = 1 \ldots |\mathcal{S}|, f = 1 \ldots |\mathcal{F}|, i = 1 \ldots |\mathcal{I}|, j = 1 \ldots |\mathcal{I}| : \mathrm{i} < \mathrm{j}, j' = 0 \ldots |\mathcal{I}| : \mathrm{j}' < \mathrm{i}$. Inactive inlet propagation.
- $l_i^{s,f} \Rightarrow \sum_{\substack{j=0 \\ \mathrm{Ind}(j)<\mathrm{Ind}(i)}}^{|\mathcal{I}|} p_{j,i}^{s,f} = 1, \forall s = 1 \ldots |\mathcal{S}|, f = 1 \ldots |\mathcal{F}|, i = 1 \ldots |\mathcal{I}|$. Exactly one follow down inlet.
- $(s_j \wedge f_k) \Rightarrow \sum_{i=1}^{|\mathcal{I}|} b_i^{j,k} \geq 1, \forall j = 1 \ldots |\mathcal{S}|, f = 1 \ldots |\mathcal{F}|$. At least one active inlet for active sector and fraction.
- $f_{f'} \Rightarrow (v_{tr} = |v - v'|^+)$. Speed contribution to transitory energy when fraction to be transported remains the same.
- $\neg f_{f'} \Rightarrow (v_{tr} = v)$. Speed contribution to transitory energy when fraction to be transported changes.
- $L_i^f < \epsilon_i^f \Rightarrow \neg b_i^f, \forall i = 1 \ldots |\mathcal{I}|, f = 1 \ldots |\mathcal{F}|$. Don't empty inlets below a residual threshold.

Objective function

Our objective function is to minimize the energy consumption plus the penalty for leaving unloaded inlets above a threshold of their maximum capacity, i.e.

$$\min\left(E_{tr} + E_{st} + P\right).$$

Observe that an optimal solution for minimizing only the energy ($E_{tr}+E_{st}$) would be obviously a solution with no inlets selected. So, the inclusion of the penalty cost associated with the no selection of overloaded inlets is essential in the model.

7. Results and Discussion

We present here empirical results when solving instances of the problem. Considering that, in a real scenario, the time duration of an emptying sequence is in the order of a few minutes, the ability of the solver to give an optimal solution in a few minutes will determine its suitability for real time operations. With this objective in mind, we have encoded three existing plants as five real problems

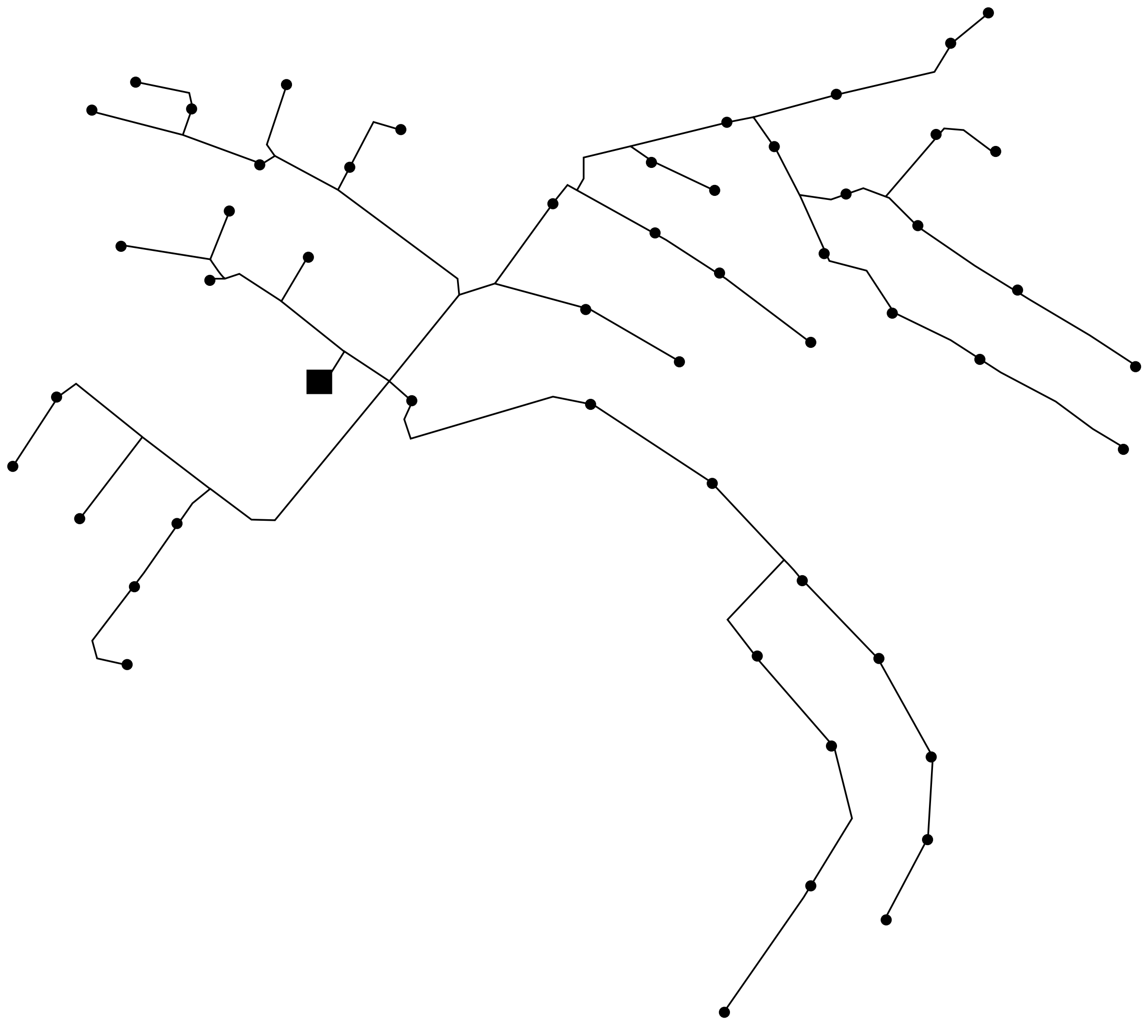

Figure 6: Schema of a real world plant (experiments 3.1 and 3.2)

and we have computed optimal solutions under different load situations. On such plant, the one labeled 3.x in our experimentation has the shape shown on Figure 6, with dots representing drop off points and the bigger square representing the central collection point.

Table 1 summarizes the results, where each entry consists of:

- A problem name with its corresponding topology. Problems p2.1 and p2.2, as well as problems p3.1 and p3.2, correspond to the same plant with different sector configurations.
- The number of drop off points and inlets.
- The number of sections and fractions.
- The number of variables and constraints before and after presolving. The SCIP [3] solver includes a series of techniques for simplifying and reducing the number of variables before starting the search.
- Time to solve. We have employed a time limit of 5 minutes. This is a typical value for the emptying sequence time on a vacuum plant and gives the time limit for taking the next decision meanwhile an emptying sequence is being operated.
- The solution gap as the relative difference between the primal and dual bounds. This gives a measure about the accuracy of the solution obtained so far in relation to the optimal solution, for those instances that were not solved within the time limit.
- The system load as the percentage of inlets with a load above the residual threshold (ϵ_i^f). This parameter increases the number of variables and constraints after the presolving phase. Among the loaded inlets, roughly a 20% of them are loaded above the penalty threshold (th_i^f).

The problems were solved with SCIP version 2.1.1[12] with SoPlex 1.6 and default settings in a 2.66 GHz processor. We selected SCIP because it provides a compact and natural encoding of our problem, and has been successfully applied to solve other challenging problems [13, 28, 36]. SCIP employs a branch-and-bound approach, which is used in all three areas, MIP, CP, and SAT. This is complemented by LP relaxations and cutting plane separators as they can be found in

[3] SCIP is short for Solving Constraint Integer Programs, see http://scip.zib.de/

MIP, by constraint specific domain propagation algorithms as in CP and SAT, and by conflict analysis as in modern SAT solvers.

It is important to note that system loads have not been tested beyond 50%. In real systems, it is a common practice to put the residual threshold around 30-40%. In other words, inlets below these occupancy levels are not selected for discharging. This makes systems loads above 50% rarely occur during a slot time.

The three studied collection plants present particular topology characteristics that allow to comment some interesting performance differences. Plant 1 is a small/medium facility, with few sector valves (3) whose all possible combinations give us 6 different sectors. As we analyze all the possible sector configurations, some of them have a large number of inlets. In particular, a sector contains all the inlets (54), and a second sector contains 45 out of the 54 total inlets. Plant 2 is a large topology, mostly lineal and highly sectored (14 sector valves). Such a large number of sector valves defines 291 possible distinct sectors, but those sector valves are located having in mind a smaller number of sectors, basically those resulting in branching from the main pipe. Even in that sector configuration, some sectors have some inlets in common, resulting in a large number of inlets on some sectors. As an example, problem p2.1 has 4 sectors, one of them having as much as 63 inlets, meanwhile problem p2.2 with 9 sectors, still presents some sectors with almost half the number of inlets (46). Plant 3 is a larger facility with a more accurate sectoring map. Its star shape topology allows to define 4 sectors (p3.1) with few intersections. Actually, only one branch is common for the 4 sectors, consisting the largest sector on 65 inlets. As an extension (p3.2), we have added a sector valve to define 5 non-overlapping sectors, reducing the largest sector to 47 inlets.

To conclude, our encoding is able to find the optimal solution for every load level on plants 1 and 3 in less than 5 minutes. A great performance improvement is observed on plant 3 (p3.2) when a better sectoring configuration is defined and every inlet belongs to exactly one sector. With respect to plant 2, no optimal solutions are found above the 20% of load, even though good solutions are found with a small gap between the primal and dual bounds. That will allow the system to cancel the search before an emptying sequence operation reaches the end, and prepare for a new emptying sequence with a good sub-optimal decision.

7.1. Software and data availability

All the set of files (`topology.xml`, `parameters.xml`, `data.xml`) used in the experiments are available to download on the web at **http://newmatica.udl.cat/**

Table 1: Solving times for three real plants. #d, #in, #s and #f stand for number of drop points, inlets, sectors and fractions, respectively

Prob.	#d	#in	#s	#f	# vars.		# const.		Time (s.)	Gap (%)	Load (%)
p1	39	54	6	2	26,910	94	35,443	264	84	0	10
p1	39	54	6	2	26,899	223	35,439	657	84	0	20
p1	39	54	6	2	26,910	513	35,434	1,413	88	0	30
p1	39	54	6	2	26,910	1,367	35,424	3,712	146	0	50
p2.1	30	102	4	4	21,708	119	28,661	336	7	0	10
p2.1	30	102	4	4	21,708	334	28,651	962	-	0.77	20
p2.1	30	102	4	4	21,708	679	28,640	1,941	-	1.45	30
p2.1	30	102	4	4	21,708	1,331	28,618	3,742	-	5.95	50
p2.2	30	102	9	4	32,086	198	42,835	568	16	0	10
p2.2	30	102	9	4	32,086	545	42,375	1,616	-	0.32	20
p2.2	30	102	9	4	32,086	1,134	42,364	3,261	-	2.38	30
p2.2	30	102	9	4	32,086	2,096	42,342	5,951	-	10.34	50
p3.1	53	109	4	2	40,126	89	52,836	211	135	0	10
p3.1	53	109	4	2	40,126	260	52,825	625	138	0	20
p3.1	53	109	4	2	40,126	639	52,816	1,500	158	0	30
p3.1	53	109	4	2	40,126	1,015	52,794	2,429	169	0	50
p3.2	53	109	5	2	15,549	49	20,540	133	23	0	10
p3.2	53	109	5	2	15,549	104	20,529	288	23	0	20
p3.2	53	109	5	2	15,549	284	20,520	746	24	0	30
p3.2	53	109	5	2	15,549	1476	20,488	1,235	25	0	50

with detailed descriptions of XML file formats and contact information with authors in case more data or clarifications on data is needed. More files, specially real world dump data logged from existing systems, will be available eventually.

Solver SCIP is available to download on the web at **http://scip.zib.de/**. The software is available with its source code, and is distributed without charge for research purposes.

8. Conclusions and future work

The collection of waste poses a major challenge on modern urban planning. The merge of information and communication technologies with traditional infrastructures, allows a smart city face the increase in waste generation and its sustainable management. AVWC reduces greenhouse gas emissions and the inconveniences of conventional methods (odours, noise, etc.). Its combination with information technologies has lead to making more efficient its energy consumption by defining smarter daily operation procedures.

The above detailed and encoded problem, along its solution, represents a single step in the quest for an optimal operation plan over an extended time horizon T. As an automated vacuum waste collection plant operates nonstop, and being a strong requirement that all the inlets must be emptied at least once a day, one can think on planning operations with a time horizon of a day. The results we have presented show that with our CIP encoding based solving approach we can find an optimal emptying sequence in a few minutes, so it is a feasible solving solution taking into account the operation times of real plants. Currently, we are working in dynamic programming techniques to determine the optimal decisions at each time slot along a complete operative time horizon. The time granularity to consider will be lower bounded by the required time to take these decisions, and consequently, determined by the performance of our CIP encoding based solving approach. So, the solving approach we have presented here will be a fundamental part of the algorithm that works over the extended time horizon, allowing an efficient learning based on historical inlet disposal data, as well as an optimal, or near-optimal, real time decision algorithm.

As then number of estates required to describe the system at each time slot is enormous, even aggregating features, we are implementing approximate dynamic programming techniques in order to perform an efficient off-line learning. As mentioned, such techniques encode the system as a CIP problem at each time slot, and solve the problem in order to determine the best decision. After several iterations, the established policies tend to be optimal, finding the most efficient emptying sequences for a day long operation.

9. Acknowledgments

This work has been partially funded by projects: ARINF (TIN2009-14704-C03-01/03) and TASSAT (TIN2010-20967-C04-01/03) from Spain MICINN and project Newmatica (IPT-2011-1496-310000) from program INNPACTO funded by MICINN (until 2011) and MINECO (from 2011).

List of acronyms and symbols

AVWC: automated vacuum waste collection.
$c_{i,t}^{tr}$: constants of transitory time.
$c_{i,t}^{st}$: constants of stationary time.
$c_{i,e}^{tr}$: constants of transitory energy.
$c_{i,e}^{st}$: constants of stationary energy.
CIP: constraint integer programming.
CP: constraint programming.
$d(\mathcal{T})$: length of tree $\mathcal{T}$.
E_t^{tr}: transitory energy.
E_t^{st}: stationary energy.
$\mathcal{E}$: set of edges.
$\mathcal{E}_t^{f,s}$: emptying sequence of sector s and fraction f at time t.
f: fraction.
$f_c(t)$: energy cost function.
$\mathcal{F}$: set of fractions.
$\mathcal{I}$: set of inlets.
I_i^f: inlet number i with fraction f.
L_i^f: load of inlet I_i^f.
L_{max}^f: maximum transfer capacity for fraction f.
$next(I_i^f)$: the following element to I_i^f in the ordered sequence $\mathcal{E}_t^{f,s}$.
s: sector.
SAT: propositional satisfiability problem.
t: time.
T_t: operation time.
T_t^{tr}: transitory operation time.
T_t^{st}: stationary operation time.

$\mathcal{T}_i^A$: air subtree.
$\mathcal{T}_i^E$: emptying subtree.
$\mathcal{T}_s^V$: vacuum subtree.
v_i^a: air valve number i.
$\mathcal{V}^a$: set of air valves.
v_i^s: sector valve number i.
$\mathcal{V}^s$: set of sector valves.
v_t: air speed during an emptying sequence.
V_M: maximum air speed.

References

[1] Achterberg, T., 2009. SCIP: Solving constraint integer programs. Mathematical Programming Computation 1 (1), 1–41.

[2] Achterberg, T., Berthold, T., Koch, T., Wolter, K., 2008. Constraint integer programming: A new approach to integrate CP and MIP. In: 5th International Conference on Integration of AI and OR Techniques in Constraint Programming, CPAIOR 2008, Paris, France, LNCS 5015. pp. 6–20.

[3] Ahcterberg, T., 2007. Constraint Integer Programming. Ph.D. thesis, Mathematik und Naturwissenschaften der Technischen Universität Berlin.

[4] Ak, E., Khan, A., 2009. A Pilot Study for the Optimization of Routes for Waste Collection Vehicles for the Göçmenköy District of Lefkoúa. Engineering and Technology, 1055–1058.

[5] Angelelli, E., Speranza, M., 2002. The application of a vehicle routing model to a waste collection problem: Two case studies. In: Klose, A., Speranza, M., Wassenhove, L. (Eds.), Quantitative Approaches to Distribution Logistics and Supply Chain Management. Vol. 519 of Lecture Notes in Economics and Mathematical Systems. Springer Berlin Heidelberg, pp. 269–286.

[6] Angelelli, E., Speranza, M. G., March 2002. The periodic vehicle routing problem with intermediate facilities. European Journal of Operational Research 137 (2), 233–247.
URL http://ideas.repec.org/a/eee/ejores/v137y2002i2p233-247.html

[7] Batty, M., Axhausen, K., Giannotti, F., Pozdnoukhov, A., Bazzani, A., Wachowicz, M., Ouzounis, G., Portugali, Y., 2012. Smart cities of the future. The European Physical Journal Special Topics 214 (1), 481–518.
URL `http://dx.doi.org/10.1140/epjst/e2012-01703-3`

[8] Bautista, J., Sep. 2008. Solving an urban waste collection problem using ants heuristics. Computers & Operations Research 35 (9), 3020–3033.
URL `http://linkinghub.elsevier.com/retrieve/pii/S0305054807000196`

[9] Béjar, R., Fernández, C., Manyà, F., Mateu, C., Sole-Mauri, F., 2012. Optimizing energy consumption in automated vacuum waste collection systems. In: Proceedings of ICTAI 2012. IEEE Press, pp. –.

[10] Béjar, R., Fernández, C., Mateu, C., Manyà, F., Sole-Mauri, F., Vidal, D., 2012. The automated vacuum waste collection optimization problem. In: Proceedings of AAAI 2012. AAAI Press / The MIT Press, pp. 264–266.

[11] Benjamin, A. M., Beasley, J. E., Apr. 2010. Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities. Computers & Operations Research 37 (12), 2270–2280.
URL `http://linkinghub.elsevier.com/retrieve/pii/S0305054810000730`

[12] Berline, Z. I., 2012. SCIP, solving constraint integer programs. `http://scip.zib.de`.

[13] Berthold, T., Heinz, S., Lübbecke, M. E., Möhring, R. H., Schulz, J., 2010. A constraint integer programming approach for resource-constrained project scheduling. In: 7th International Conference on Integration of AI and OR Techniques in Constraint Programming, CPAIOR 2010, Bologna, Italy, LNCS 6140. pp. 51–55.

[14] Eisenstein, D. D., Iyer, A. V., 1997. Garbage collection in chicago: A dynamic scheduling model. Management Science 43 (7), 922–933.
URL `http://mansci.journal.informs.org/content/43/7/922.abstract`

[15] Eurostat, European Comission, Dec. 2012. Waste statistics. http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Waste_statistics.

[16] Gaube, V., Remesch, A., 2013. Impact of urban planning on household's residential decisions: An agent-based simulation model for Vienna. Environmental Modelling & Software In press (In press), –.
URL http://www.sciencedirect.com/science/article/pii/S1364815212002836

[17] Guariso, G., Michetti, F., Porta, F., Moore, S., Nov. 2009. Modelling the upgrade of an urban waste disposal system. Environ. Model. Softw. 24 (11), 1314–1322.
URL http://dx.doi.org/10.1016/j.envsoft.2009.04.008

[18] Hancke, G. P., Silva, B. d. C. e., Hancke, Jr., G. P., 2012. The role of advanced sensing in smart cities. Sensors 13 (1), 393–425.
URL http://www.mdpi.com/1424-8220/13/1/393

[19] Hilty, L. M., Arnfalk, P., Erdmann, L., Goodman, J., Lehmann, M., Wäger, P. A., 2006. The relevance of information and communication technologies for environmental sustainability – A prospective simulation study. Environmental Modelling & Software 21 (11), 1618–1629.

[20] Kim, B., Kim, S., Sahoo, S., Dec. 2006. Waste collection vehicle routing problem with time windows. Computers & Operations Research 33 (12), 3624–3642.
URL http://linkinghub.elsevier.com/retrieve/pii/S0305054805001322

[21] Larsen, A., Vrgoc, M., Christensen, T., Lieberknecht, P., 2009. Diesel consumption in waste collection and transport and its environmental significance. Waste Management and Research 27 (7), 652–659.

[22] Lazaroiu, G. C., Roscia, M., 2012. Definition methodology for the smart cities model. Energy 47 (1), 326 – 332.
URL http://www.sciencedirect.com/science/article/pii/S0360544212007062

[23] Lopes, T. M. T., Ciré, A. A., de Souza, C. C., Moura, A. V., 2010. A hybrid model for a multiproduct pipeline planning and scheduling problem. Constraints 15 (2), 151–189.

[24] Moura, A. V., de Souza, C. C., Ciré, A. A., Lopes, T. M. T., 2008. Planning and scheduling the operation of a very large oil pipeline network. In: Principles and Practice of Constraint Programming, 14th International Conference, (CP 2008). pp. 36–51.

[25] Nijkamp, P., Kourtit, K., 2013. The 'New Urban Europe': Global challenges and local responses in the urban century. European Planning Studies 21 (3), 291–315.
URL `http://www.tandfonline.com/doi/abs/10.1080/09654313.2012.716243`

[26] O'Grady, M., O'Hare, G., 2012. How smart is your city? Science 335 (6076), 1581–1582.
URL `http://www.sciencemag.org/content/335/6076/1581.short`

[27] O'Sullivan, B., 2012. Opportunities and challenges for constraint programming. In: Proceedings of AAAI 2012. AAAI Press / The MIT Press, pp. 2148–2152.
URL `http://www.aaai.org/ocs/index.php/AAAI/AAAI12/paper/view/5189`

[28] Pfetsch, M. E., 2008. Branch-and-cut for the maximum feasible subsystem problem. SIAM Journal on Optimization 19 (1), 21–38.

[29] Poot, A., Kant, G., Wagelmans, A., January 2002. A savings based method for real-life vehicle routing problems. Operational Research Society. Journal, 57–68.
URL `http://dx.doi.org/10.1057=palgrave=jors=2601252`

[30] Potvin, J.-Y., Rousseau, J.-M., 1993. A parallel route building algorithm for the vehicle routing and scheduling problem with time windows. European Journal of Operational Research 66 (3), 331 – 340.
URL `http://www.sciencedirect.com/science/article/pii/037722179390221S`

[31] Rochat, Y., Taillard, R. D., 1995. Probabilistic diversification and intensification in local search for vehicle routing. Journal of Heuristics 1, 147–167.

[32] Solomon, M. M., 1987. Algorithms for the vehicle routing and scheduling problems with time window constraints. Operations Research 35, 254–265.

[33] Taillard, E., Badeau, P., Gendreau, M., Guertin, F., Potvin, J.-Y., 1997. A tabu search heuristic for the vehicle routing problem with soft time windows. Transportation Science 31 (2), 170–186.
URL http://transci.journal.informs.org/content/31/2/170.abstract

[34] Teixeira, J., Antunes, A. P., de Sousa, J. P., 2004. Recyclable waste collection planning – A case study. European Journal of Operational Research 158 (3), 543 – 554.
URL http://www.sciencedirect.com/science/article/pii/S0377221703003795

[35] Tung, D., Pinnoi, A., Sep. 2000. Vehicle routing-scheduling for waste collection in Hanoi. European Journal of Operational Research 125 (3), 449–468.
URL http://linkinghub.elsevier.com/retrieve/pii/S0377221799004087

[36] van der Veen, E., Veltman, B., 2009. Rostering from staffing levels: a branch-and-price approach. In: 35th International Conference on Operational Research Applied to Health Services. pp. 1–10.

[37] Worrell, W. A., Vesilind, P. A., 2012. Solid Waste Engineering, 2nd Edition. Cengage Learning, pp. 116–117.

[38] Xiong, Z., Sheng, H., RONGWenGe, Cooper, D. E., 2012. Intelligent transportation systems for smart cities: a progress review. Science China Information Sciences 55 (12), 2908.